VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

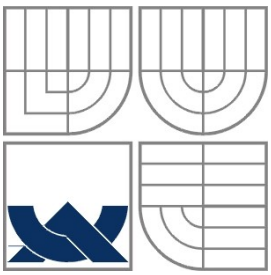
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

AUTOMORFING DVOU OBRÁZKŮ

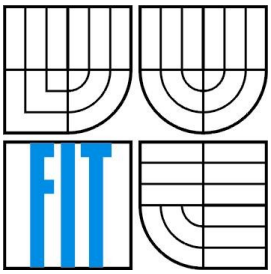
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE PAVEL ČERMÁK
AUTHOR

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

AUTOMORFING DVOU OBRÁZKŮ

AUTOMORPHING OF THE IMAGE PAIRS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

PAVEL ČERMÁK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. VÍTĚZSLAV BERAN, Ph.D.

BRNO 2011

Abstrakt

Tato práce se zabývá tvorbou automorfingu mezi zdrojovým a cílovým obrazem. Práce popisuje již existující metody a algoritmy pro tvorbu morfingu. Dále se práce zaměřuje na popis návrhu a realizaci metody pro tvorbu automorfingu. K tomuto účelu je zapotřebí detekovat významné body v obraze, tyto body korespondovat a podle těchto korespondencí vytvořit vlastní morfing.

Abstract

This thesis I deal with creating automorphing between source and destination image. The thesis describes existing methods and algorithms for creating morphing. The thesis focuses on the design and implementation methods for creating automorphing. For this purpose it is necessary to detect significant points in the image, find correspondence of to these points and create your own morphing.

Klíčová slova

Morfing, warping, Delaunayho triangulace, SURF, afinní transformace, interpolace, klíčový bod, deskriptor.

Keywords

Morphing, warping, Delaunay triangulation, SURF, affine transformation, interpolation, keypoint, descriptor.

Citace

Pavel Čermák: Automorfing dvou obrázků, bakalářská práce, Brno, FIT VUT v Brně, 2011

Automorfing dvou obrázků

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Vítězslava Berana Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Pavel Čermák
18. května 2011

Poděkování

Rád bych poděkoval vedoucímu mé práce Ing. Vítězslavu Beranovi Ph.D. za obětavost, metodické rady a konstruktivní kritiku při řešení a posléze i psaní bakalářské práce.

Obsah

1 Úvod

Tato práce se zabývá technikou morfing, která vytváří plynulý přechod mezi dvěma obrázky. Úkolem morfingu je nalézt mezi zdrojovým a cílovým obrazem takovou transformaci, která působí reálným dojmem. Většina již navrženým aplikací, které se zabývají technikou morfing, pracují za předpokladu, že uživatel zadá sám body, podle kterých se bude morfing provádět. Cílem této práce bude vytvořit aplikaci, která bude automaticky detekovat klíčové body v obraze a podle nich pak vytvářet morfing mezi dvěma obrázky.

Úvodní část práce tvoří kapitola, která má čtenáře seznámit s různými technikami a možnostmi jak morfing realizovat. Jsou zde popsány základní metody, které jsou pro morfing důležité. Mezi tyto metody zejména patří warping, interpolace nebo také afinní transformace obrazu. Jelikož se práce zaměřuje na automatickou detekci bodů v obraze, jsou zde i uvedeny různé algoritmy pro detekci významných bodů v obraze.

Další kapitolou této práce je již samotný návrh aplikace. Jak se zde čtenář může dočíst, aplikace se skládá z pěti základních bloků , které jsou v této kapitole podrobněji rozebrány. Mezi tyto základní bloky patří předzpracování obrazu, detekce bodů, korespondence bodů, vytvoření sítě, a vlastní morfing dvou obrázků. Jelikož pro samotný morfing nebyla využita žádná knihovna, která by realizovala tento blok návrhu, jsou zde i popsány dva algoritmy, které popisují jak je v aplikaci realizován warping a morfing obrazu.

Třetí kapitola práce je rozdělena do dvou částí. V první části je popsána realizace aplikace. Obsahem této části je mimo jiné vymezení vstupních a výstupních dat, se kterými aplikace pracuje. Dále je zde napsán manuál k vytvořené aplikaci. Druhá část této kapitoly je věnována experimentům s navrženou aplikací. Celkem jsou zde popsány dva experimenty. První experiment zjišťuje časovou náročnost morfingu pro různé obrazy. Další experiment je zaměřen spíše na vizuální vzhled daných přechodů mezi obrázky. Tyto přechody jsou poté porovnávány s výstupy od jiných volně dostupných aplikací.

Poslední kapitolou je samotný závěr, který shrnuje dosažené výsledky a naznačuje možné vylepšení do budoucnosti.

2 Vymezení základních pojmů

Obsahem této kapitoly bude vysvětlení základních pojmů týkajících se morfingu a detekce významných bodů v obraze. Čtenář se zde seznámí s různými možnostmi jak morfing realizovat.

2.1 Morfing

Morfing je obecně přeměna čehokoliv v cokoliv. Jedná se o proces, který očekává na vstupu dva bitmapové obrázky a jejich topologické informace o způsobu tvarové změny v obrázku. Výstupem morfingu je animovaná sekvence obrázků, kde lze pozorovat postupnou změnu zdrojového obrázku na cílový obrázek. Nejdůležitější částí tohoto procesu je definice topologické informace tvarové změny. V tom nejjednodušším případě je morfing tvořen pouze prolínáním zdrojového a cílového obrázku. Přičemž z počátku je výsledný obraz tvořen z větší části informací ze zdrojového obrazu. Během animace se pak ubírá ze zdrojového obrazu a přidává se informace z obrazu cílového. Tento postup lze aplikovat na obrazy s podobným geometrickým uspořádáním. V praxi se tento postup moc nepoužívá, protože je malá pravděpodobnost, že oba obrazy budou přibližně stejné. Proto se v morfingu používá tzv. warping, který zajišťuje požadované tvarové změny v obraze podle jejich topologického popisu (převzato z [8]).

Historie morfingu [8] nesahá příliš hluboko do minulosti. Za jedno z prvních využití morfingu je označován klip Michaela Jacksona k písni Black or White z roku 1991. V té době se jednalo o převratný trik, který obdivovaly stovky tisíc lidí a morfing tím vstoupil do povědomí široké veřejnosti. Prvním filmem, ve kterém byla využita řada pokročilých morfovacích technik se stal Terminátor 2, který je z roku 1992. V dnešní době se již nikdo nepozastaví nad těmito triky, které se staly běžnou součástí filmů, klipů a reklam. Od této doby se morfing příliš nezměnil. Změnila se pouze výpočetní technika, která dovoluje počítačům vytvářet čím dál tím více realistické iluze.

Morfing se v širokém množství používá v televizi, reklamách a filmech, kde se postavy mění v jiné nebo jakoby zmizí ze scény [8]. Zároveň se morfing uplatnil i v jiných netradičních oborech. V soudním lékařství napomáhá vědcům k rekonstrukci tvaru obličeje ze znalosti lidské lebky a informacích o obličejové tkáni. Další uplatnění našel morfing v aplikacích pro umělé vytváření fotografií. Tohoto uplatnění morfingu využívá hojně policie, která může takto určit vzhled pohřešované osoby za určitý čas.

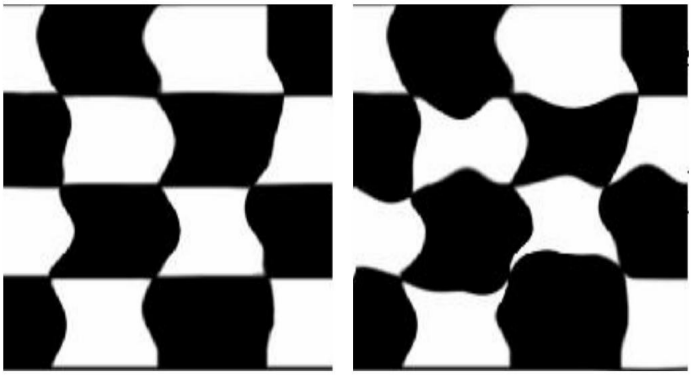
Obr. 2.1 Příklad výstupu metodou morfing (převzato z [8])

2.2 Dvouprůchodový síťový warping

Jedná se o algoritmus, který byl vytvořen roku 1990 Doublasem Smythem [7]. Tento algoritmus je specifický tím, že na zdrojový i cílový obraz je přichycena síť, jejíž editací na cílovém obraze lze provádět požadované deformace. Jelikož v čase definice sítě ještě neznáme cílový obraz, volíme jeho obsah buď prázdný nebo jej zkopírujeme ze zdrojového obrazu. Jednotlivé uzly sítě jsou propojeny křivkami (v praxi se běžně jedná o přímky, Beziériovy křivky, B-spline nebo Catmul-Rom). Tyto křivky jsou popsány matematickými rovnicemi. Minimální počet uzlů, které je potřeba definovat, aby algoritmus pracoval správně je 4. Je vhodné volit dostatečný počet bodů sítě, aby deformace byla jemná, ale na druhou stranu požadujeme rychlý výpočet. Jako rozumný počet uzlů se považuje číslo 9 a vyšší. Nutnou podmínkou této metody je, aby obě sítě byly topologicky totožné. Tímto se rozumí, aby byl stejný počet křivek jak ve vodorovném tak i svislé směru. Algoritmus pracuje ve dvou průchodech.

První průchod lze rozdělit do dvou částí. Nejprve si spočítáme průsečíky svislých křivek s vodorovnými řádky obrazu a to jak ve zdrojovém, tak i cílovém obraze. Tím pádem se nám každý řádek rozdělí do intervalů, které poté stačí převzorkovat do cílového obrazu. Prvním průchodem vytvoříme meziobraz, kde jsou na správných pozicích pixely v řádcích.

Druhý průchod má obdobný postup jako ten první. Nyní místo rozdělení řádků do intervalů rozdělíme sloupce do intervalů. Za zdrojový obraz nyní považujeme meziobraz, který jsme vytvořili v prvním průchodu. Intervaly ve sloupcích jsou určeny vodorovnými křivkami v obou obrazech. Nyní již stačí převzorkovat dané intervaly z meziobrazu do cílového obrazu. Oba průchody jsou znázorněny na obrázku 2.2.



Obr. 2.2 Šachovnice po prvním průchodu (vlevo) a po druhém průchodu (převzato z [7])

Nevýhodou této metody je, že ji lze použít pouze na globální zpracování obrazu. Toto je způsobeno proto, že se počítá s celou sítí na původním obrázku. Pokud daná síť nebude příliš hustá, budou se deformovat velké plochy, které nezaručí uspokojivý výsledek. Na druhou stranu při velkém množství uzlů se zvyšuje čas výpočtu samotného algoritmu.

2.3 Warping pomocí korespondujících úseček

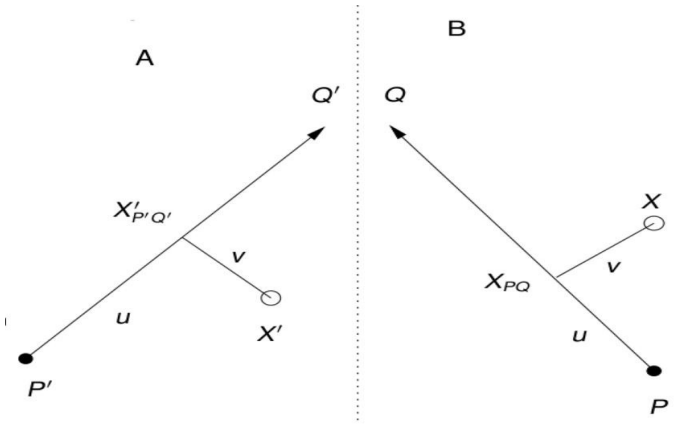
Další z řady algoritmů [4] založených na nelineárním zpracování obrazu, je warping pomocí korespondujících si úseček. Tato metoda je založena na jiném principu než byl síťový warping, a proto se používá tam, kde zmíněný algoritmus neuspěl. Jeho použití je především tam, kde chceme provádět lokální změny v obraze, jako jsou zavření očí, otočení objektu, posunutí jeho částí apod.

Základ algoritmu tvoří korespondující si úsečky, které jsou vkládány do obou obrázků. Dané úsečky svou velikostí a polohou definují lokální změnu v obraze. Cílovým obrazem se zde rozumí kopie původního obrazu.

U tohoto algoritmu se využívá tzv. zpětné mapování, které zaručuje, že v cílovém obraze bude každý pixel zastoupen nějakou hodnotou ze zdrojového obrazu. Existuje ještě varianta tzv. dopředného mapování, které ale nezaručuje, že každý pixel v cílovém obraze bude namapován na některou hodnotou z obrazu zdrojového. Veškeré vzorce v této podkapitole jsou převzaty z [4].

2.3.1 Transformace jedním párem úseček

Uvažujme situaci, která je na obrázku 2.3. Na obrázku vidíme dvě úsečky, úsečka *PQ*, která leží v cílovém obraze a úsečka *P´Q´*, která leží ve zdrojovém obraze. Předpokládejme že body *P*, *Q*, *P´ a Q´* jsou body obrazu, a proto mají celočíselné souřadnice. Postup je jednoduchý, budeme procházet každý pixel cílového obrazu a k němu budeme hledat jeho vzor ve zdrojovém obrazu (na obrázku naznačeno body *X* a *X'*). Bod *X'*, který je bodem zdrojové obrazu, nemá celočíselné souřadnice, a proto je potřeba tento bod interpolovat některou z metod (více o interpolaci lze nalézt v kapitole 2.6).



Obr. 2.3 Transformace jedním párem úseček (převzato z [4])

Jak je patrné z obrázku 2.3, je potřeba nejprve vypočítat koeficienty *u* a *v*, které určují polohu pixelu *X* vzhledem k úsečce *PQ*, ale zároveň i určují polohu bodu *X´* vzhledem k úsečce *P´Q´*.

$$u = \frac{(X-P)*(Q-P)}{\|Q-P\|^2} \tag{2.1}$$

Analogicky se může spočítat koeficient *v*:

$$v = \frac{(X-P)*(Q-P)^{\perp}}{\|Q-P\|} \tag{2.2}$$

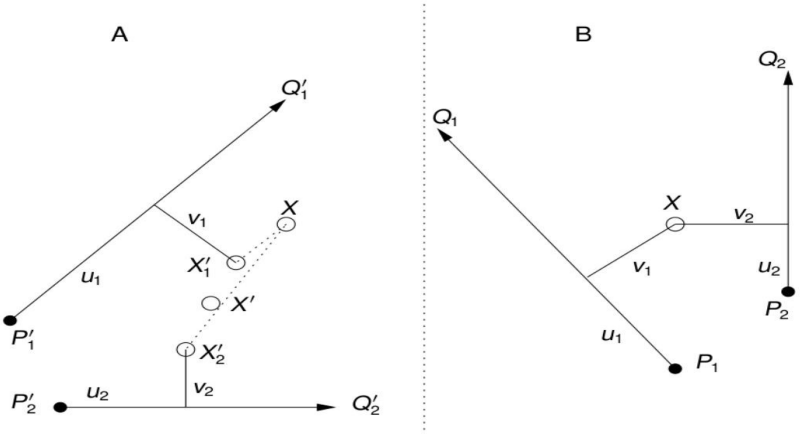
kde vektor $v^{\perp} = (-y, x)$ je kolmý k vektoru $v = (x, y)$.

Souřadnice bodu *X'* pak spočítáme snadno jako

$$X' = P' + u*(Q'-P') + \frac{v*(Q'-P')^{\perp}}{\|Q'-P'\|} \tag{2.3}$$

2.3.2 Transformace s více páry úseček

Pokud použijeme pouze jeden pár úseček, je přiřazení bodu ze zdrojového obrazu do cílového jednoznačné. Toto ovšem neplatí o transformaci s více páry úseček. Mějme schéma, které je na obrázku 2.4.



Obr. 2.4 Warping pomocí více párů úseček (převzato z [4])

Použijeme-li více úseček, pak pro bod X v cílovém obraze nalezneme hodnoty u a v pro každou úsečku v tomto obraze. Zde nastává odlišnost od předchozího postupu, protože korespondující úsečky ve zdrojovém obraze vytváří množinu bodů X_i', a proto se hodnota X vypočítá jako vážený součet všech hodnot X_i'. Je-li definována množina úseček P_iQ_i v cílovém obraze a množina korespondujících úseček $P'_iQ'_i$ v obraze zdrojovém, pak můžeme podle vztahu (2.1) a (2.2) vypočítat hodnoty u_i a v_i a souřadnice X_i na základě vzorce

$$X = \sum_{i=1}^{n} w_i * X'_i \tag{2.4}$$

kde w_i je váha příspěvku X_i a vypočítáme ji z rovnice

$$w_i = \frac{|P_i * Q_i|}{|P_i * Q_i| + dist^2} \tag{2.5}$$

kde *dist* je vzdálenost bodu X od úsečky P_iQ_i určena předpisem

$$dist = \begin{cases} |v| \ldots . \; pro \; u \in \langle 0,1 \rangle \\ ||P - X|| \ldots . \; pro \; u < 0 \\ ||Q - X|| \ldots . \; pro \; u > 1 \end{cases} \tag{2.6}$$

2.4 Deformace obrazu pomocí trojúhelníků

Poslední algoritmus, který zde uvedu, používaným pro warping, je deformace pomocí trojúhelníků [7]. Základem algoritmu je rozdělení obrazu na trojúhelníky, které se posléze deformují. Dělení plochy obrazu do trojúhelníků se zpravidla provádí dvěma způsoby:

- **ručně** – nejprve definujeme v obraze řídící body, a poté označíme takové trojice řídících bodů, které budou tvořit trojúhelníky.
- **automaticky** – opět rozmístíme řídící body v obraze, a poté se využije některého z algoritmů pro vytvoření trojúhelníků.

Poté co rozdělíme plochu zdrojového a cílového obrazu pomocí trojúhelníků, nám potřebné tvarové změny zajistí posuny řídících bodů ve zdrojové obrazu.

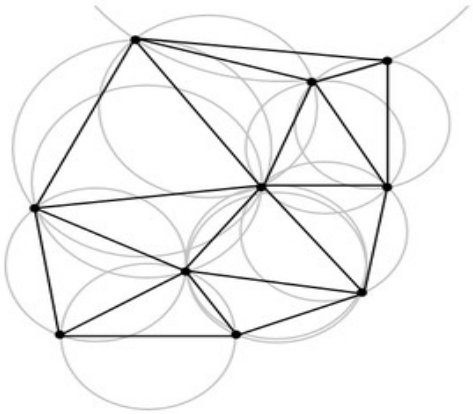
Velkou výhodou této metody je, že zde nenastává problém nekonvexních útvarů jako v případě čtyřúhelníků, protože trojúhelníky jsou vždy konvexní. Jednou z nevýhod tohoto řešení je, že pokud chceme dosáhnout uspokojivých výsledků je potřeba vytvořit i velké množství trojúhelníků, což je výpočetně náročnější. Pro vytvoření triangulační sítě se nejčastěji používá Delaunayho tringulace.

2.4.1 Delaunayho triangulace

Jedná se o algoritmus, který slouží k popisu objektů pomocí trojúhelníků. Byl vynalezen Borisem Delaunay v roce 1934 [5]. Tento algoritmus je základem naprosté většiny automatických algoritmů pro vytváření trojúhelníkových sítí. Základem algoritmu je popsat obraz pomocí jednotlivých trojúhelníků.

Nechť P je množina n bodů v rovině, které neleží na přímce a k je počet bodů, které leží na hranici konvexního obalu bodů z množiny P. Pak platí, že každá Delaunayho triangulace z množiny P má právě $2*n - 2 - k$ trojúhelníků a $3*n - 3 - k$ hran (převzato ze zdroje [5]).

Podmínkou triangulace je, že každá opsaná kružnice v Delaunayho triangulaci nesmí obsahovat žádné další body z množiny P. Navíc každý trojúhelník by měl být co možná nejvíce rovnostranný.



Obr. 2.5 Vytvoření Delaunayho triangulace pro 10 bodů (převzato z [5])

Celý algoritmus [7] lze popsat následovně. Nejprve se vytvoří tzv. supertrojúhelník, který obsahuje všechny vkládané body (dopředu musí být jasné, na jak velké ploše se budou body vkládat). Nyní postupně vkládáme jednotlivé body, u kterých kontrolujeme splnění podmínek triangulace. Tyto body následně mění stávající trojúhelníky v triangulaci nebo vytvářejí nové. Pro každý přidaný bod se provede následující kontrola:

- zkontroluje se platnost všech podmínek triangulace pro všechny existující trojúhelníky.
- ty trojúhelníky, pro které neplatí, se rozloží na jednotlivé hrany.
- duplicitně popsané hrany jsou hranami vnitřními a zruší se.
- ze zbylých hran a vkládaného bodu se vytvoří nové trojúhelníky.

Jako poslední krok algoritmu se provede odstranění všech trojúhelníků, které obsahují body původního supertrojúhelníku.

2.5 Afinní transformace

Jedná se o transformace [3], které jsou nejčastěji používané v počítačové grafice. Mezi základní afinní transformace patří posunutí, otáčení, změna měřítka, zkosení a další operace, které vzniknou kombinací již zmíněných operací.

Obecně lze afinní transformaci vyjádřit vztahem $P' = P * A$, kde P a P' jsou body zdrojového a cílového obrazu, přičemž bod P transformujeme pomocí matice A na bod P'. Matice A zde představuje jednotlivé transformační operace.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = A * \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\begin{aligned} x' &= A_{00} + A_{01}x + A_{02}y \\ y' &= A_{10} + A_{11}x + A_{12}y \end{aligned} \tag{2.7}$$

2.5.1 Skládání transformací

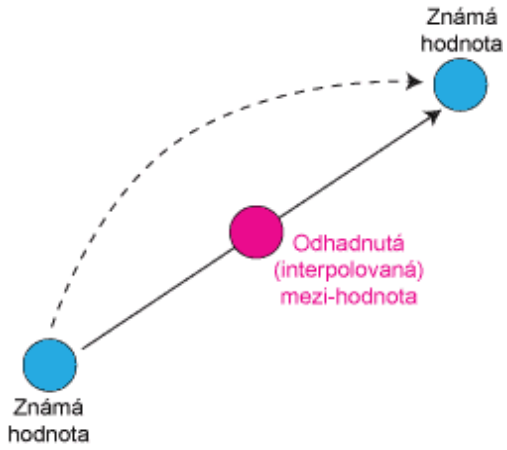
Jak už název napovídá tento druh transformace se skládá s dílčích transformací. Pokud aplikujeme jednotlivé transformace na bod *P* záleží přitom na pořadí provádění transformací. Je velký rozdíl pokud bychom nejprve na bod *P* aplikovaly transformaci posunutí a poté transformaci otáčení nebo naopak. Transformace, která vznikla složením jednotlivých dílčích transformací lze vyjádřit pomocí jedné matice. Tuto matici dostaneme pokud dané dílčí transformační matice vynásobíme mezi sebou. Jelikož záleží na pořadí aplikování transformací na objekt, záleží také na pořadí násobení transformačních matic.

2.5.2 Homogenní souřadnice

Myšlenkou zavedení homogenních souřadnic je reprezentace bodu ve vektorovém prostoru o jednu dimenzi větším. Čili daný prostor rozšíříme o jednu dimenzi. Bod *P[X,Y]* lze v homogenních souřadnicích zapsat jako *P[X,Y,W]*, kde $w \neq 0$. Nejčastěji volíme homogenní souřadnici $w = 1$. Převod bodu *P[X,Y,W]* v homogenních souřadnicích lze jednoduše převést do kartézských souřadnic pomocí vztahu $X_K = X/W$ a $Y_K = Y/W$. Homogenní souřadnice nám umožňují pracovat se všemi transformacemi pomocí maticového zápisu.

2.6 Interpolace

Pod pojmem interpolace [9] si lze představit metodu pro rekonstrukci obrazu, která určuje hodnotu obrazu v místě, kde doposud není známa. Jedná se tedy o metodu, která se snaží nalézt přibližné hodnoty funkce v daném intervalu ze známých hodnot okolních bodů (viz. obrázek 2.6). Matematickým výpočtem lze poté odhadnout přibližnou hodnotu tohoto neznámého místa.

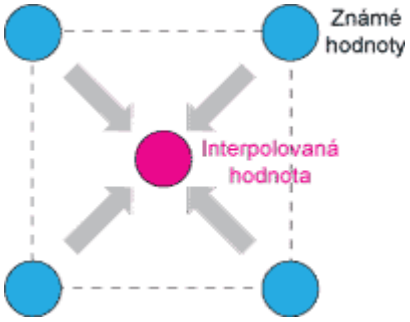


Obr. 2.6 Interpolace bodu (převzato z [10])

Je velké množství metod, které se zabývají interpolací obrazu [9]. Základní dělení těchto metod je na algoritmy neadaptivní a adaptivní. Neadaptivní metody (např. metoda nejbližšího souseda, lineární, bilineární atd.) pracují nad celým obrazem bez ohledu na obsah daného obrazu. Naopak adaptivní algoritmy se snaží pochopit topologii daného obrazu a pro každou rozdílnou část obrazu zvolit nejlepší interpolační metodu. Text níže blíže popisuje nejznámější interpolační techniky.

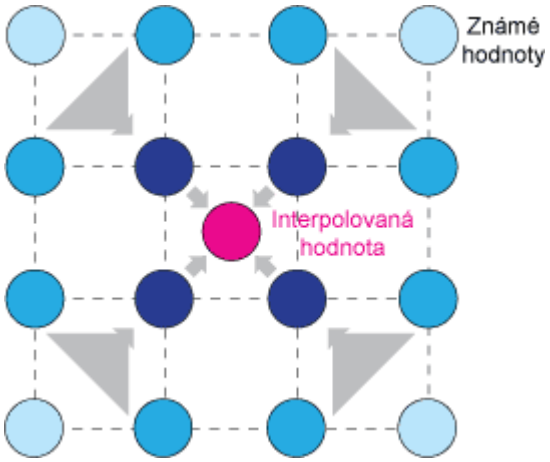
Nejjednodušším způsobem interpolace je interpolace nejbližším sousedem [10]. Hodnotou nejbližšího souseda se myslí bod v obraze, který je od požadované hodnoty nejméně vzdálen. Celou metodu lze implementovat jako obyčejné zaokrouhlení dané hodnoty.

Technika bilineární interpolace [10] používá čtyři nejbližší sousední body (okolí bodu 2x2) a z těchto bodů poté počítá váženým průměrem nový pixel (viz. obrázek 2.7). Metoda je mnohem jemnější než při použití metody nejbližšího souseda. Problém této metody nastává při zvětšování snímku, kde lze pozorovat viditelné rozostření.



Obr. 2.7 Bilineární interpolace (převzato z [10])

Poslední významnou technikou je bikubická interpolace [10]. Jedná se o polynomiální interpolaci třetího stupně. Výpočet se provádí pomocí kubického polynomu. Celkem se pro bikubickou interpolaci používá šestnáct pixelů (viz obrázek 2.8). Největším rozdílem oproti předchozí metodě je v tom, že zde se místo přímky používá kubická křivka zadaná čtyřmi body v jedné dimenzi. Je zjevné, že tato metoda je pomalejší než předchozí metody.



Obr. 2.8 Bikubická interpolace (převzato z [10])

2.7 Detekce významných bodů

V této podkapitole budou rozebrány metody pro detekci významných bodů v obraze. Jelikož pro detekci bodů využívám metody SURF a GoodFeaturesToTrack, zaměřím se v této podkapitole právě na tyto metody.

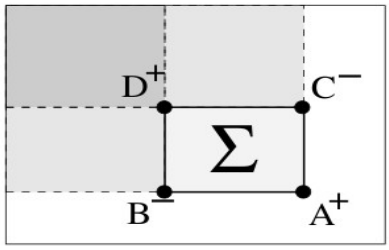
2.7.1 SURF (Speeded Up Robust Features)

Jedná se o relativně novou metodu [11], kterou roku 2006 publikovali H. Bay, T. Tuytelaars a Luc Van Gool, která vyhledává a koresponduje významné body v obraze. Tato metoda vychází z metody SIFT, od které se liší pouze aproximací složitých a pomalých prostředků. Postup při detekci a výpočtu deskriptoru klíčového bodu lze shrnout do 3 částí.

Pro vyhledání klíčových bodů v obraze se používá konvoluce obrazu s jádry derivovanými od Gaussovy funkce. Pro rychlejší detekci klíčových bodů se používá výpočet determinantu Hessianovy matice , ve kterém se využívá výhod integrálního obrazu.

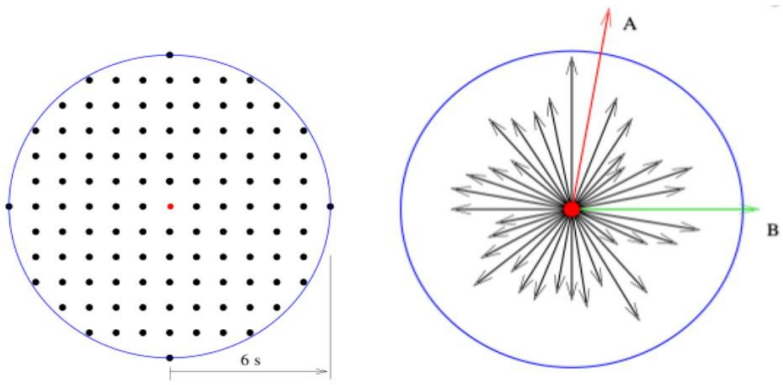
Integrální obraz je struktura vybudovaná nad zdrojovým obrazem, kde každý bod integrálního obrazu se spočítá jako součet hodnot původního obrazu ohraničený levým horním rohem obrazu a

souřadnicemi bodu, který je počítán. Výpočet každého bodu integrálního obrazu je výpočetně nenáročné, protože se při výpočtu bodu používají body, které jsou již vypočtené (viz. obrázek 2.9).



Obr. 2.9 Integrální obraz, kde A, B, C, D jsou body obrazu (převzato z [6])

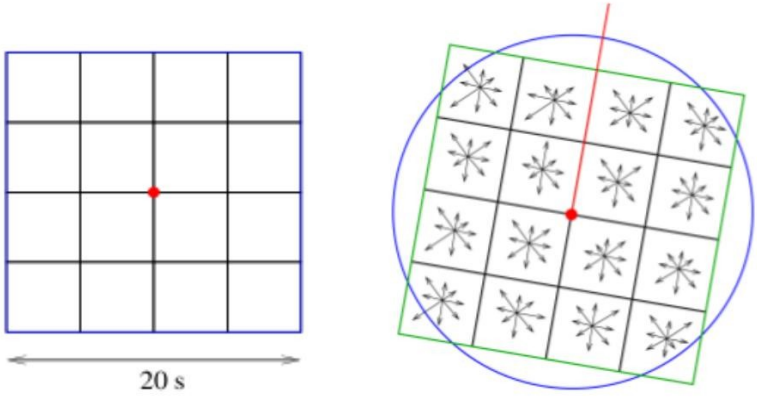
Pro zajištění nezávislosti metody na změně polohy obrazu je klíčovým bodům přidělena orientace. V kruhovém okolí bodu o poloměru *6s*, kde s je měřítko bodu, se spočítá pro každý bod gradient tohoto bodu (*dx*, *dy*). Jako struktura pro záznam orientací bodů se používají posuvná okénka. Každé okénko v sobě uchovává odděleně součet hodnot *dx* a *dy*. Příslušnost daného bodu do určitého okénka se určí na základě gradientu tohoto bodu z hodnot *dx* a *dy*. Po zjištění všech hodnot gradientů daných bodů tvoří výsledné součty v jednotlivých okénkách nové vektory. Tyto vektory se mezi sebou porovnají a vybere se z nich ten největší. Orientaci klíčového bodu určuje právě orientace vektoru s největší hodnotou (viz obrázek 2.10).



Obr. 2.10 Vlevo okolí zkoumaného bodu a vpravo histogram orientací okolních bodů (převzato z [6])

Pro výpočet deskriptoru se pro každý klíčový bod sestrojí čtvercová oblast o straně *20s*, která je natočená podle orientace klíčového bodu. Celá oblast se rozdělí na podoblasti o velikosti *5s* (viz obrázek 2.11). V každé této oblasti se stanový 5 bodů, na které se aplikuje filtr pro výpočet gradientu (velikost *2s*). Každá podoblast se zpracovává samostatně. Pro každý bod z podoblasti jsou vypočítány

hodnoty dx a dy a přičteny k danému subdeskriptoru, který je tvořen z hodnot (*suma(dx)*, *suma(|dx|)*, *suma(dy)*, *suma(|dy|)*). Tyto 4 hodnoty se získají z každé podoblasti. Výsledný deskriptor je tvořen hodnotami ze všech podoblastí. Velikost výsledného deskriptoru může být buďto 64 nebo 128 binů.



Obr. 2.11 Vlevo rozdělení okolí bodu do oblastí a vpravo histogram deskriptorů v jednotlivých oblastech (převzato z [6])

Metodu SURF lze využít v různých oblastech počítačového vidění. Tuto metodu lze použít k rychlému popisu obrázků pomocí deskriptorů, rekonstrukci 2D a 3D scén nebo klasifikaci obrázků.

2.7.2 GoodFeaturesToTrack

Jedná se o metodu, která detekuje nejvýraznější body v obraze. Algoritmus vymysleli Jianbo Shi a Carlo Tomasi a poprvé ho prezentovali roku 1994 na konferenci počítačového vidění [12]. Hlavním úkolem algoritmu bylo nalézt dostatečně velkou množinu kvalitních bodů vhodných pro sledování pohybu mezi dvěma obrazy. Algoritmus musel být schopen najít výrazné body v různých vstupních obrázcích bez znalosti jejich vlastností. Počet nalezených bodů by se měla pohybovat v rozumných mezích, aby nevznikaly příliš velké výpočetní nároky při detekci bodů.

Princip algoritmus vychází z Harrisonova-Stephensova detektoru rohů. Upravuje rovnici pro výpočet indikátoru přítomnosti rohu *R* na:

$$R = min(\lambda 1, \lambda 2) \tag{2.8}$$

kde $\lambda 1$ a $\lambda 2$ představují vlastní čísla, která odráží hladkost okolí zkoumaného bodu a podle těchto parametrů lze poznat , jestli je v daném bodě roh a jak je významný.

Jak je vidět parametr *R* uchovává menší z hodnot vlastních čísel. Pokud tato menší hodnota vlastního čísla vykazuje velkou odezvu (větší než práh), potom i hrana v tomto místě bývá významná.

Jako hodnota *R* v definovaném okolí bodu se uchovává menší hodnota z vlastních čísel. Tento parametr *R* se přiřadí každému pixelu obrazu. Zároveň je i uchováván největší parametr *R* v celém obraze. Následně se vyberou takové body obrazu podle zadaného koeficientu kvality, které mají lepší mez zadanou jako poměr k nejlepšímu bodu (např, 0,1 čili 10%). Počet detekovaných bodů je dále omezen hodnotou pro maximální počet bodů v obraze (*Nmax*). Ve výsledku to znamená, že se v obraze detekují pouze dostatečně kvalitní body. Jako poslední se odstraní body, které jsou si blíže než je zadaná minimální hodnota.

2.8 Volně dostupné aplikace pro morfing

Tato podkapitola popisuje různé aplikace, které se zabývají metodou morfingu obrázků. Z volně dostupných aplikací jsem vybral FantaMorph, SmartMorph a FotoMorph.

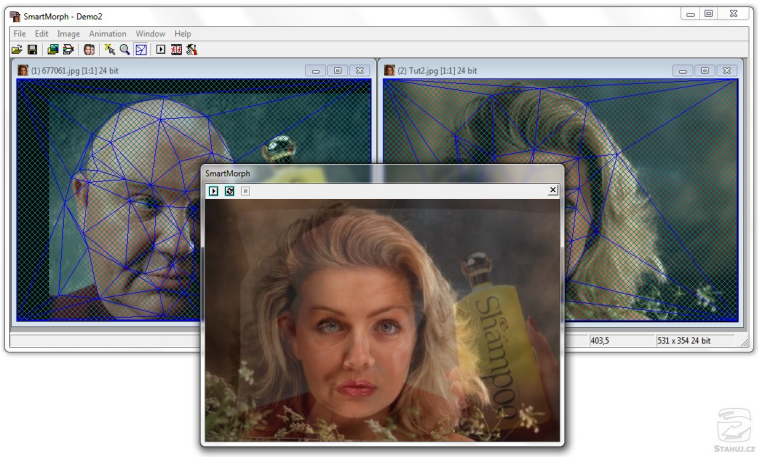
2.8.1 FantaMorph

FantaMorph [13] je aplikace, která umožňuje provádět plynulé přechody mezi jednotlivými obrázky. Dané animace umožňuje exportovat do sekvencí obrázků, videa nebo animovaného GIFU. Mezi jeho výhody patří intuitivní ovládání a příjemné grafické prostředí. Jelikož aplikace vyžaduje po uživateli zadávání klíčových bodů obrázku, bude výsledný morfing vždy podle představ uživatele. Vzhled aplikace FantaMorph je vidět na obrázku 2.12.

Obr. 2.12 Obrázek aplikace FantaMorph (převzato z [13])

2.8.2 SmartMorph

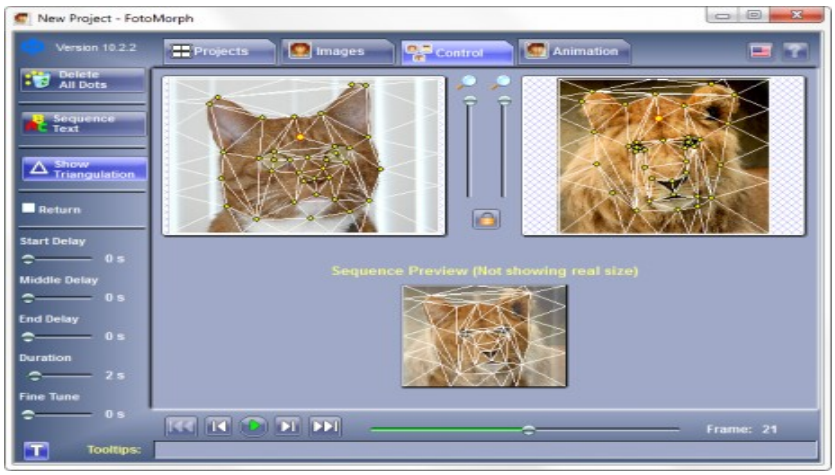
SmartMorph [15] je animační software, který umožňuje provádět nad obrázky metody morfing a warping. Aplikace dále umožňuje provádět s obrázkem různé jasové transformace pomocí filtrů. Uživatel musí opět pro vytvoření plynulého přechodu zadat klíčové body v každém obrázku. Jako většina těchto aplikací je i SmartMorph distribuován pod free licencí. Vzhled aplikace SmartMorph je na obrázku 2.13



Obr. 2.13 Obrázek aplikace SmartMorph (převzato z [15])

2.8.3 FotoMorph

FotoMorph je program [14], který je určen na tvorbu obrazových animací, ve kterém dochází k plynulému přechodu mezi jednotlivými obrázky (morfing). Aplikace umožňuje provádět metodu warping nad jedním obrázkem. Nad importovanými obrázky lze provádět různé deformace pomocí různých filtrů. Danou animaci dovoluje exportovat do sekvencí obrázků JPEG, animovaného GIFU nebo flash videa. Vytvoření a korespondenci bodů si zde musí uživatel určit sám. Aplikace má příjemné uživatelské rozhraní a lehké ovládání, které je patrné na obrázku 2.14.

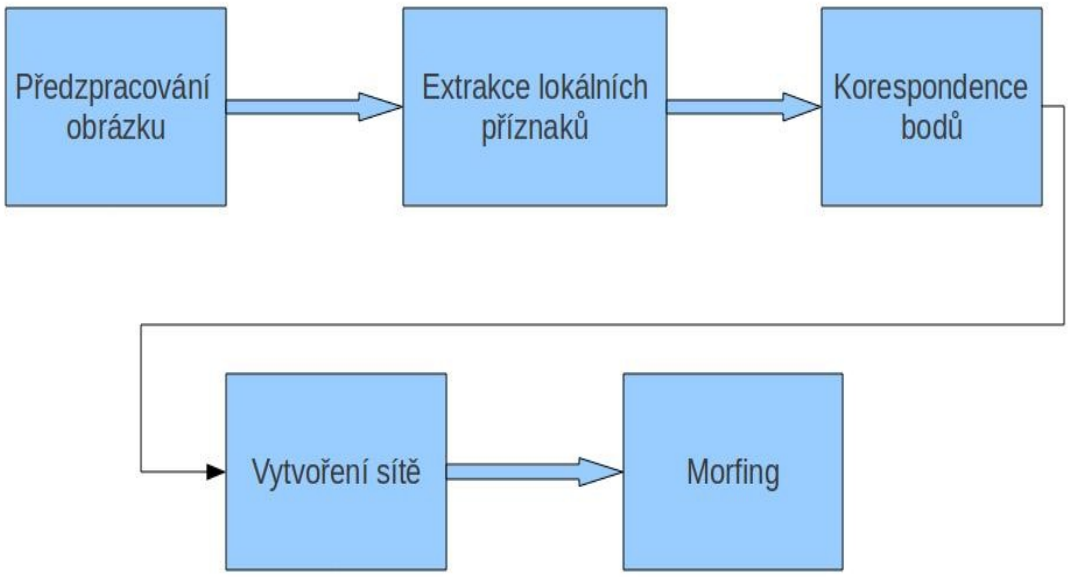


Obr. 2.14 Obrázek aplikace FotoMorph (převzato z [14])

Následující přehled ukazuje různé aplikace, které se metodou morfingu zabývají. Zmíněné aplikace však od uživatele vyžadují zadávání klíčových bodů. Proto cílem této práce je realizovat aplikaci, která bude metodu morfing provádět bez zásahu uživatele.

3 Návrh aplikace

V této kapitole popíši vlastní návrh řešení, který jsem použil k vytvoření demonstrující aplikace. Cílem mého snažení je vytvořit aplikaci, která bude automaticky detekovat body v obraze a pomocí těchto bodů bude provádět metodu morfing mezi zdrojovým a cílovým obrazem. Cílem je, aby uživatel nemusel zadávat klíčové body obrazu sám, ale aby se celý proces provedl automaticky. K tomuto účelu jsem vytvořil návrh aplikace, který jsem rozdělil do 5 částí. Mezi hlavní části návrhu patří detekce bodů, jejich korespondence, vytvoření triangulační sítě a samotný morfing dvou obrázků. Pro detekci bodů jsem využil algoritmus GoodFeaturesToTrack. Pro korespondenci bodů jsem si vytvořil matici korespondecí, která určuje míru podobnosti daných bodů. Na vytvoření triangulační sítě jsem použil algoritmus Delaunayho triangulace, který vytvoří automaticky triangulační síť z detekovaných bodů. Metodu morfing jsem realizoval jako kombinaci warpingu obrazů a jejich prolnutí. Výsledné schéma návrhu je na obrázku 3.1.



Obr. 3.1 Schéma návrhu aplikace

3.1 Předzpracování obrázku

Prvním blokem návrhu aplikace je předzpracování obrázku. Pokud chci s obrázkem pracovat musím ho nejprve převést do takové formy, abych s ním mohl pracovat v dalších blocích. První úpravou obrázků (zdrojového i cílového) je jejich převod do šedotónových obrazů. Tento převod je důležitý kvůli zvolenému algoritmu na vyhledávání zájmových bodů v obraze. Další úpravou vstupních obrazů je změna jejich velikostí. Tato úprava je zavedena proto, aby aplikace pracovala se stejně velkými obrazy. Vždy se změní ten obraz, který má větší plochu.

3.2 Extrakce lokálních příznaků

Dalším blokem návrhu je vyhledání klíčových bodů v obraze. Abych mohl dané obrázky morfovat, musím v každém z nich vyhledat určité body zájmu, podle kterých provedu jejich morfing.

Mým první pokusem k vyhledání klíčových bodů v obraze bylo rozdělení obrazu (zdrojového i cílového) na jednotlivé segmenty, kde daný segment tvořili body se stejnou nebo alespoň podobnou intenzitou barvy. Hledaný klíčový bod se pak nacházel na průsečíku daných segmentů. Toto řešení se ale ukázalo jako nevyhovující kvůli nerovnoměrnému rozložení klíčových bodů v obraze. Další nevýhodou bylo to, že klíčové body se hledaly pouze na základě barevné informace v bodech. Proto jsem své řešení obměnil a využil jsem jednu z existujících metod na vyhledávání klíčových bodů v obraze.

Algoritmus, který jsem použil se jmenuje SURF (Speeded Up Robust Features). Jeho výhodou oproti mému předešlému řešení bylo v tom, že algoritmus při vyhledání klíčových bodů nepracuje jen s barevnou informací bodu, ale i s topologií obrazu. Jeho největší výhodou je možnost ke každému vyhledanému bodu vytvořit jeho deskriptor, který jsem poté použil k vytvoření korespondencí mezi zdrojovými a cílovými klíčovými body obou obrazů . Toto řešení se z počátku zdálo vyhovující, ale později jsem zjistil, že pro automorfing bude potřeba nadetekovat více klíčových bodů.

Jako poslední a zároveň finální verzi tohoto bloku jsem místo algoritmu SURF použil algoritmus GoodFeaturesToTrack, který se více zaměřuje při hledání klíčových bodů na hrany v obraze. Z původního návrhu jsem nechal pouze SURF popis klíčových bodů (SURF deskriptor).

3.3 Korespondence bodů

Jelikož vytvářím aplikaci, která automaticky detekuje významné body v obraze, je nutné těmto bodům určit korespondence. K tomuto účelu jsem si vytvořil tzv. matici korespondencí, ve které každý prvek matice určuje míru podobnosti daných párů bodů. Abych mohl detekované body korespondovat, musím si určit podle jakých kritérií budu korespondenci provádět. Z předešlé podkapitoly je zřejmé, že jedním kritériem korespondencí bude korespondence pomocí deskriptorů daných bodů. Mezi další kritéria, která jsem zařadil do korespondence bodů, patří Euklidovská vzdálenost bodů a barevná informace v těchto bodech obsažená. Souhrnně lze tedy říct, že pro korespondenci jednoho páru bodů budu počítat metriky (vzdálenosti) ve třech základních prostorech (deskriptorový, obrazový a barevný).

Mezi první kritérium korespondence jsem zařadil korespondenci pomocí deskriptorů bodů. Pro tento účel jsem využil SURF deskriptory bodů. Deskriptor lze chápat jako desetinné číslo a jejich podobnost *l* lze určit na základě vztahu:

$$l=\sqrt{\sum_{i=0}^{127}(desc1_i-desc2_i)^2} \tag{3.1}$$

kde *desc1* a *desc2* jsou deskriptory bodů obrazu.

Druhým zmiňovaným kritériem je Euklidovská vzdálenost dvou bodů v obraze. Jelikož pracuji ve dvourozměrném prostoru, tak pro výpočet vzdálenosti *d* mezi body *X[x1,x2]* a *Y[y1,y2]* v obrazovém prostoru platí rovnice:

$$d=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2} \tag{3.2}$$

Posledním kritériem pro korespondenci bodů je jejich barevná informace. Podobnost *p* daných barev *color1 = (r1,b1,g1)* a *color2 = (r2,b2,g2)* lze určit na základě odečtu jednotlivých barevných složek. Lze tedy napsat:

$$p=((|r1-r2|)/255+(|b1-b2|)/255+(|g1-g2|)/255)/3 \tag{3.3}$$

Výsledkem tohoto vztahu je průměrná odchylka daných dvou barev.

Posledním krokem tohoto bloku je přiřazení vah jednotlivým metrikám. Tento krok jsem zavedl, abych určil, která metrika je pro mě důležitá a která ne. Proto jsem jednotlivým metrikám přidělil následující váhy:

- **deskriptor** - 0,4
- **euklidovská vzdálenost** - 0,5
- **barevná informace** – 0,1

3.4 Vytvoření sítě

Neméně důležitým blokem návrhu je vytvoření sítě z detekovaných bodů. K tomuto účelu jsem použil již existující algoritmus, který automaticky vytvoří triangulační síť z detekovaných bodů. Tento algoritmus se jmenuje Delaunayho triangulace (více v kapitole 2.4.1). Tato síť se vytvoří na zdrojovém i cílovém obraze a výsledné meziobrazy se poté tvoří posouváním bodů v těchto sítích.

3.5 Morfing

Posledním blokem návrhu je vlastní morfing dvou obrázků. Jak je patrné z podkapitoly 2.1, existují dva druhy morfingu. Rozhodl jsem se pro variantu prolnutí + warping, která působí realističtějším dojmem. Vlastní algoritmy pro warping a morfing jsou detailněji popsány níže.

3.5.1 Algoritmus morfingu

Pro vytvoření morfingu mezi dvěma obrazy jsem zvolil variantu, která kombinuje warping a prolínání obrazu.

Algoritmus očekává na vstupu zdrojový a cílový obraz, množiny bodů těchto obrazů a definované trojúhelníky v těchto obrazech. Jako posledním parametrem algoritmu je počet obrazů, který má výsledný morfing obsahovat. Princip algoritmu je následující. Pro každý meziobraz si vypočítám množinu posunutých bodů mezi zdrojovými a cílovými body. Množinu posunutých bodů získám jako rozdíl mezi zdrojovým a cílovým bodem v obou osách (x-ová i y-ová) a následným dělením těchto rozdílovým hodnot celkovým počtem obrazů, které má morfing vytvořit. Poté už jenom stačí k daným zdrojovým bodům příčíst přírustek v obou osách, který vznikl tímto dělením. Dále si od zdrojového a cílového obrazu vytvořím warpované obrazy podle vypočítané množiny posunutých bodů. Poté si vytvořím tzv. váhu obrazu, která určuje jaký obraz v daném meziobraze převládá. Nakonec provedu prolnutí těchto dvou obrazů a zobrazím výsledný obraz.

Realizace techniky prolnutí provádím jako přiřazení koeficientu průhlednosti každému pixelu obrazu. Tento koeficient si označím písmenem α. Koeficient nabývá průhlednosti v rozmezí <0,1>.

Hodnota nula obvykle znamená dokonale neprůhledný pixel a maximální hodnota dokonale průhledný pixel. Koeficient průhlednost lze také chápat jako procento pokrytí daného pixelu barvou. Prolnutí obrazu pak počítám na základě rovnice

$$I = I_A * \alpha + I_B * (1 - \alpha) \tag{3.4}$$

kde I je výsledná hodnota intenzity pixelu, I_A a I_B jsou vstupní pixely obrazů a α je koeficient průhlednosti.

3.5.2 Algoritmus warpingu

Pro vytvoření warpingu obrazu bylo důležité znát posunutí bodů, podle kterých se warping provede. Dále jsem potřeboval určit mapovací funkci mezi originálním a warpovaným obrazem. Neméně důležitou částí byla interpolace bodů získaných z mapovací funkce.

Algoritmus očekává na vstupu obraz určený k warpingu, dále pak seznam trojúhelníků definovaných v obraze a zdrojové a posunuté body obrazu. Algoritmus prochází trojúhelníky obrazu a ke každému trojúhelníku nalezne odpovídající si trojúhelníky ve warpovaném obraze. Zde využívám toho, že množiny zdrojových a posunutých bodů jsou v korespondenci. Čili, že první položce ze zdrojových bodů odpovídá první položka z bodů posunutých. Pokud tedy znám zdrojové trojúhelníky obrazu, lehce k nim naleznu odpovídající si warpované trojúhelníky. Pro každé korespondující si trojúhelníky si vytvořím mapovací funkci pomocí afinní transformace, která zajistí namapování zdrojového trojúhelníku na warpovaný trojúhelník. Pokud již znám mapovací funkci, můžu pro každý pixel warpovaného trojúhelníku určit jeho hodnotu ve zdrojovém trojúhelníku tzv. zpětné mapování. Jelikož mapovací funkce transformuje body tak, že nemusejí zapadnout do obrazové mřížky, je potřeba tyto body interpolovat. K tomuto účelu jsem využil interpolaci metodou nejbližšího souseda, kterou lze implementovat jako prosté zaokrouhlení daného bodu. Tento postup pak opakuji pro každý trojúhelník obrazu. Výsledkem algoritmu je pak warpovaný obraz, který je pak dále zpracováván algoritmem pro morfing.

4 Realizace a experimenty

Celý program je napsaný v jazyce C/C++. Pro implementaci jsem využil knihovnu OpenCV, která je volně šiřitelná knihovna počítačového vidění. Byla vyvinuta společností Intel. Pracuje pod operačními systémy Windows, Linux, Mac OS a je napsaná v jazyce C/C++. Tato knihovna obsahuje velké množství funkcí a metod, které pracují s obrázky a videi. Je šířena pod BSD licencí.

4.1 Vstupní a výstupní data

Vstupem programu jsou dva obrazy určené k morfování. Jelikož jsem využil knihovnu OpenCV, je formát obrázků omezen pouze na formáty podporované touto knihovnou. Mezi tyto formáty patří BMP, GIF, JPEG, JPG,

Předpokládá se, že pokud má být morfing co nejvíce realistický je potřeba aby zdrojový a cílový obraz měly stejný ústřední objekt (např. obličeje, motorky, auta). Pokud jsou zadány dva topologicky odlišné obrazy nelze očekávat realistický přechod mezi obrazy.

Program umožňuje zpracovávat obrazy, které nemají stejné rozměry. Pokud jsou takovéto obrazy zadány, program provede zmenšení toho obrazu, který má větší rozměry.

Výstupem programu je sekvence obrázků, které vytvářejí plynulý přechod mezi zdrojovým a cílovým obrazem. Tuto sekvenci si lze prohlížet při běhu aplikace pomocí posuvníku nebo si lze danou sekvenci vyexportovat do formátu JPG.

4.2 Manipulace s aplikací Automorph

Čtenáře zde seznámím s možnostmi implementované aplikace Automorph a jejím ovládáním.

Program se neinstaluje, je spustitelný přímo z CD. Načítání obou vstupních obrazů se děje přímo při spuštění aplikace. Názvy obou obrazů se zadávají jako parametry příkazové řádky. Při spuštění programu je nutné, aby se program spustil nejméně se dvěma parametry a to *-s* a *-d*.

Zadávání parametrů

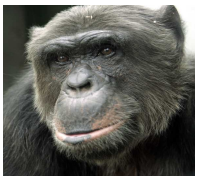
Parametry lze zadávat v libovolném pořadí.
./automorph -s <source image> -d <destination image> -t -f <count frames> -w

-s <source image> - parametr pro načtení zdrojového obrázku
-d <destination image> - parametr pro načtení cílového obrázku
-t - je-li parametr zadán, zobrazí se navíc i triangulační síť obou obrázků
-f <count frames> - tímto parametrem lze zadat výsledný počet obrázků morfingu (implicitně 10)
-w - tímto parametrem lze uložit výsledné obrázky morfingu

Po spuštění programu se objeví jedno až tři okna podle toho zda je zadán parametr *-t* nebo ne. Vždy se zobrazí hlavní okno, ve kterém se pomocí táhla lze pohybovat v meziobrazech vytvořených morfingem.

4.3 Měření délky trvání morfingu

Pro měření času potřebného k výpočtu morfingu byly vybrány 3 páry obrázků. Důležitou roli zde bude hrát počet detekovaných bodů a z toho plynoucí počet trojúhelníků. Čím více bodů se nalezne, tím více se vytvoří trojúhelníků a tím pádem se i prodlužuje doba trvání výsledného morfingu. V tabulce 4.1 jsou vypsány jednotlivé obrázky s konkrétními počty detekovaných bodů a trojúhelníků. Názvy obrázků jsou stejné s názvy souborů, ve kterých jsou uloženy (přiloženo na CD).

ID	Název obrázku	Rozlišení	Počet bodů	Počet trojúhelníků	Náhled
1	opice.jpg	590x640	45	68	
2	kote.jpg	400x300	28	34	
3	cyklista.jpg	600x400	85	148	
4	motorka.jpg	1920x1200	71	120	
5	tygrik.jpg	460x340	63	104	
6	mercedes.jpg	650x500	76	130	

Tab. 4.1 Tabulka detekovaných bodů a trojúhelníků v obrazech

	1 -> 2	3 -> 4	5 -> 6
čas výpočtu [s]	0,9	1,6	1,5
počet korespondujících si trojúhelníků	34	120	104

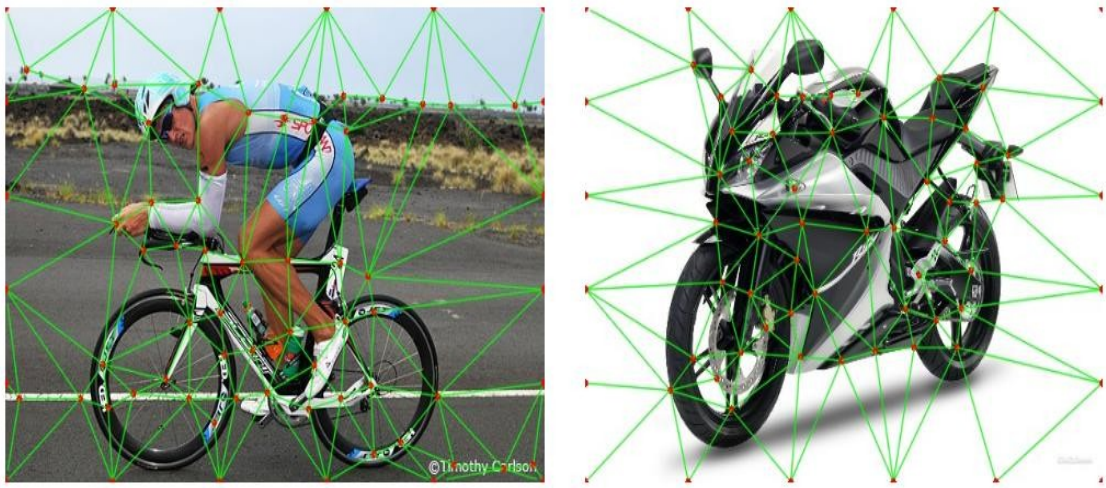
Tab. 4.2 Tabulka ukazující doby výpočtu morfingu

Z tabulky 4.2 lze vidět, že při vyšším počtu trojúhelníků se čas potřebný k výpočtu morfingu prodlužuje. Navíc lze pozorovat, že výsledný počet trojúhelníků, které se použijí při morfingu, je roven trojúhelníkům, které se detekovaly v obraze s menším počtem bodů.

Možnou optimalizací programu by mohlo být efektivnější vyhledávání trojúhelníků v Delaunayho triangulaci. Stávající řešení nalézá některé trojúhelníky opakovaně a tím prodlužuje čas výpočtu morfingu. Dalším vylepšením může být efektivnější vyhledávání korespondujících si trojúhelníků při vytváření warpingu obrazu. Stávající řešení, k vytvoření korespondujících si trojúhelníků, prochází 3x množinu zdrojových bodů obrazu (pro každý bod trojúhelníku) k nalezení korespondence.

4.4 Srovnání výstupů s jinými aplikacemi

Druhým testem je srovnání výstupu implementované metody s volně šiřitelnými aplikacemi. V těchto testech jsem využil program *image_compare* (přiložen na CD), který dokáže vyhodnotit podobnost daným obrazů. Test jsem provedl na dvou párech obrázků, které se od sebe liší ústředním objektem, který v obraze dominuje.

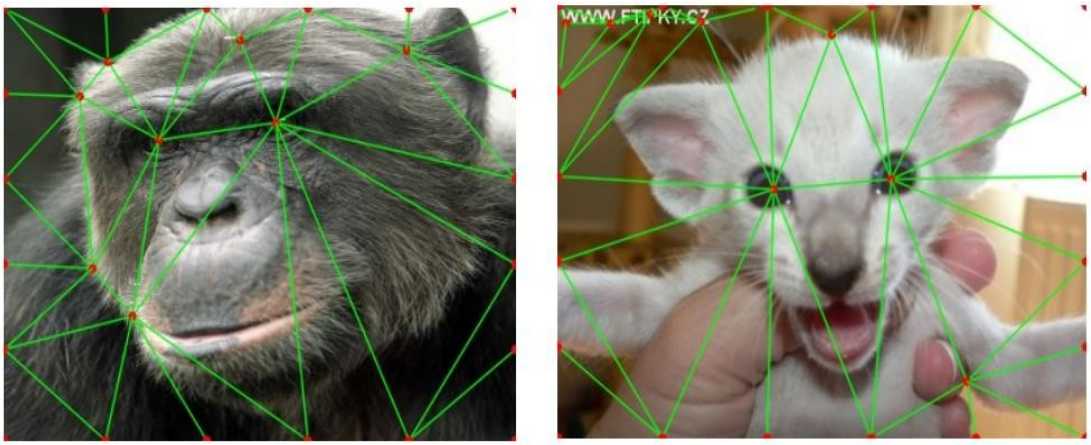


Obr. 4.1 Dealunayho tringulace pro obrázky cyklysta.jpg a motorka.jpg

Obr. 4.2 Výstup z realizované aplikace Automorph

Obr 4.3 Výstup z aplikace FotoMorph

Pomocí programu *image_compare* jsem vyhodnotil podobnost sekvencí obrázků, které jsou zobrazeny na obrázcích 4.2 a 4.3 na 89%. Do tohoto čísla jsem nezapočítal rozdíly prvních a posledních obrázků, protože jsou tyto obrázky stejné a danou celkovou podobnost by to zkreslilo.



Obr. 4.4 Dealunayho tringulace pro obrázky opice.jpg a kote.jpg

Obr. 4.5 Výstup s realizovaná aplikace Automorph

Obr 4.6 Výstup z aplikace FantaMorph

V tomto případě jsem podobnost těchto dvou sekvencí obrázků (4.5 a 4.6) vyhodnotil na 93,5%. I zde jsem nezapočítal do výsledného součtu první a poslední obrázky.

Po vizuální stránce lze usoudit, že přechody mezi obrazy působí přiměřeným realistickým dojmem. Při různých ústředních objektech obrázků, nelze pozorovat přirozený realistický dojem. Zatímco obrazy, které mají stejný nebo alespoň podobný ústřední objekt vykazují realističtější přechod mezi zdrojovým a cílovým obrazem

Všechny testy byly prováděny na 32 bitovém stroji značky Intel, který disponuje procesor s frekvencí 2.1GHz a operační pamětí 3GB. Testy jsem provedl v prostředí s operačním systémem Ubuntu 10.10.

5 Závěr

Celá práce se zabývá technikou morfingu dvou obrazů. Velká část práce je věnována popisu existujících metod pro tvorbu warpingu. Součástí práce je realizace metody, která je schopna provést automorfing dvou obrazů. Cílem mého snažení bylo vytvořit konzolovou aplikaci, která bude provádět automatickou detekci bodů v obou obrazech a z těchto bodů pak vytvoří morfing těchto obrazů.

Po podrobném prostudování základních technik, které se používají pro morfing, jsem vytvořil aplikaci Automorph, která realizuje automorfing dvou obrázků. Návrh celé aplikace jsem rozdělil do 5 základních bloků. Prvním blokem návrhu jsem převedl oba vstupní obrázky do takové formy, abych s nimi mohl pracovat v dalších krocích návrhu. Druhým blokem návrhu jsem provedl detekci významných bodů v obou obrázcích, podle kterých jsem pak provedl požadované deformace obrazu. Jeden z nejdůležitějším bloků návrhu je korespondence detekovaných významných bodů. Zde jsem ke každému detekovanému bodu vytvořil jeho deskriptor. Pro korespondenci bodů jsem vzal v úvahu metriky tří základních prostorů (deskriptorový, obrazový a barevný). Jednotlivé metriky jsem poté opatřil určitou váhou, která symbolizuje důležitost dané metriky při korespondenci bodů. Předposledním blokem návrhu bylo vytvoření sítě z nadetekovaných bodů v obrazech. Pro tento účel jsem použil algoritmus Delaunayho triangulace, který je jedním z nejčastěji používaných algoritmů pro vytváření triangulačních sítí. Posledním blokem celého návrhu pak bylo provést metodu morfingu mezi zdrojovým a cílovým obrazem.

Podle daných výsledků experimentů lze usoudit, že aplikace Automorph je vhodnější pro obrázky, které mají podobný charakter (např. obličeje, auta, motorky). Při topologicky různých obrazech (např. motorka a cyklista), lze pozorovat pouze přiměřený realistický dojem v jednotlivých obrazových přechodech. Toto vyplývá i z provedeného experimentu, který prokazuje 89% shodu meziobrazů mezi obrázky cyklista.jpg a motorka.jpg . Naopak při topologicky podobných obrázcích je shoda 93,5% (obrázky opice.jpg a kote.jpg).

Tento projekt by šlo rozšířit o několik dalších vylepšení. Základním rozšířením aplikace by mohlo být zakomponování více metod pro detekci bodů v obraze a z těchto metod poté vybírat tu nejlepší pro daný druh obrázků. Velice zajímavým rozšířením by byl převod stávajícího řešení do 3D, které by pracovalo s tělesy.

Literatura

[1] OpenCV Wiki. OpenCV 2.1 C++ manual [online].c2010.[citováno 2011-4-20]. Dostupné na WWW: <http://opencv.willowgarage.com/documentation/cpp/index.html>

[2] Bradski, Gary R; Kaehler, Adrian. Learning OpenCV. Sebastopol : O'Reilly, c2008. 555s. ISBN 978-059-6516-130.

[3] WikiKnihy. Geometrie/Afinní transformace souřadnic [online].c2010.[citováno 2011-4-20] Dostupné na WWW: <http://cs.wikibooks.org/wiki/Geometrie/Afinní_transformace_souřadnic>

[4] Thaddeus Beier, Shawn Neely. Feature-Based Image Metamorphosis [online]. SIGGRAPH 1 992.[citováno 2008-4-20],s.35-42. Dostupné na WWW: <http://www.cs.princeton.edu/courses/archive/fall00/cs426/papers/beier92.pdf>

[5] Wikipedia.org. Delaunay triangulation [online].c2011.[citováno 2011-4-10]. Dostupné na WWW: <http://en.wikipedia.org/wiki/Delaunay_triangulation>

[6] Bílek, Petr. Významné body v obraze: detekce, lokalizace a korespondence ve 3D. Praha: České vysoké učení technické, Fakulta elektrotechnická, 2007. Bakalářká práce.

[7] Dobšík, Martin. Morfing. Brno: Vysoké učení technické, Fakulta informačních technologií, 1997. Diplomová práce.

[8] Wikipedia.org. Morphing [online].c2011.[citováno 2011-4-10] Dostupné na WWW: <http://en.wikipedia.org/wiki/Morphing>

[9] Wikipedia.org. Interpolation [online].c2011.[citováno 2011-4-10] Dostupné na WWW: <http://en.wikipedia.org/wiki/Interpolation>

[10] Pihan, Roman. Interpolace [online].c2010.[citováno 2011-4-3] Dostupné na WWW: <http://www.fotoroman.cz/glossary2/3_interpolace.htm>

[11] H. Bay, T. Tuytelaars, L. van Gool. SURF: Speeded up robust features. In Proceedings of European Conference on Computer Vision, pages 404–417, 2006. Dostupné na WWW: <http://www.vision.ee.ethz.ch/publications/papers/proceedings/eth_biwi_00392.pdf>

[12] Jianbo Shi, Carlo Tomasi. Good features to track. Proc. IEEE Comput. Soc. Conf Comput. Vision and Pattern Recogn., pages 593-300,1994. Dostupné na WWW: <http://www.ai.mit.edu/courses/6.891/handouts/shi94good.pdf>

[13] Studna.cz. FantaMorph [online].c2010.[citováno 2011-5-5] Dostupné na WWW: <http://www.studna.cz/abrosoft-fantamorph-p-2759.html>

[14] Studna.cz. FotoMorph [online].c2010.[citováno 2011-5-5] Dostupné na WWW: <http://www.studna.cz/fotomorph-p-15092.html>

[15] Stahuj.centrum.cz. SmartMorph [online].c2010.[citováno 2011-5-5]
Dostupné na WWW:
<http://www.stahuj.centrum.cz/grafika_a_design/tvorba_grafiky/ostatni/smartmorph/>

Seznam příloh

Příloha 1. CD se zdrojovými texty, plakátem a sadou fotografií